LES TROPHÉES NSI

Édition 2023

DOSSIER DE CANDIDATURE
PRÉSENTATION DU PROJET

Ce document est l'un des livrables à fournir lors du dépôt de votre projet : 4 pages maximum (hors documentation).

Pour accéder à la liste complète des éléments à fournir, consultez la page **Préparer votre participation**.

Vous avez des questions sur le concours ? Vous souhaitez des informations complémentaires pour déposer un projet ? Contactez-nous à **info@trophees-nsi.fr**.

NOM DU PROJET : Pyracode

> PRÉSENTATION GÉNÉRALE :

• *Idée et objectifs*
• *Origines et intérêts du projet*

Pyracode est un jeu d'aventure et d'énigmes développé en Python à l'aide de la bibliothèque Pygame. Le joueur explore une pyramide, combat des monstres redoutables, et résout des problèmes informatiques tout en découvrant les secrets cachés de l'Egypte antique. Le but du projet était de créer un jeu qui soit à la fois ludique, avec un système de tir amusant et des designs authentiques, et éducatif, en donnant un avant-goût de la spécialité NSI grâce à des codes à trous à compléter ou des arbres binaires à résoudre.

Au cours de l'année, notre professeur nous a fait travailler en équipe sur de nombreux projets. Vers mars, nous avons décidé de participer à ce concours en présentant un projet original qui utiliserait toutes les techniques apprises, notamment la programmation orientée objet, les interfaces graphiques et la gestion des processus. Nous avons choisi de proposer ce projet pour partager notre passion pour la conception de programmes informatiques et pour montrer que la spécialité NSI est accessible à tous et qu'elle est cruciale pour les métiers du futur. Malgré les défis posés par les épreuves de spécialités en mars, nous avons réussi à rendre le projet à temps grâce aux heures de projets dédiées de cet enseignement, que nous avons principalement utilisées pour les réunions et l'assemblage des travaux de chacun. Nous avons également travaillé en dehors de l'établissement en utilisant un dossier partagé pour l'échange de fichiers et une messagerie instantanée pour la communication à distance.

> ORGANISATION DU TRAVAIL :

• *Présentation de l'équipe (prénom de chaque membre et rôle dans le projet)*
• *Répartition des tâches*
• *Organisation du travail (répartition par petits groupes, fréquence de réunions, travail en dehors de l'établissement scolaire, outils/logiciels utilisés pour la communication et le partage du code, etc.)*

Notre équipe de cinq personnes a relevé le défi ambitieux de créer un jeu de A à Z sans utiliser de moteur de jeu. Nous avons su nous organiser de manière efficace pour mener à bien ce projet, en répartissant les tâches en fonction des compétences de chacun.

Noah CUNEO et Gauthier NOEL ont pris le rôle de développeur senior dans ce projet. Ils ont su coordonner la création de la structure physique du jeu ainsi que la réalisation des énigmes et des menus avec brio. Noah a notamment géré le système de sauvegarde et le fonctionnement global du jeu. Gauthier quant à lui, a réalisé les énigmes, les menus et a apporté une touche de charme avec des musiques dans un style 8-bit grâce au logiciel Bosca Ceoil.

Nos graphistes ont été des éléments clés pour donner vie à notre univers de jeu. Léa ZUCCHI--JOURDAN a non seulement dessiné presque l'intégralité du jeu, des animations aux maps en passant par les menus et les énigmes, mais elle a également intégré ses créations dans le code du jeu. Baptiste DUVAL a quant à lui créé le pixel art détaillé du personnage ainsi que les boss, inspirés de l'Egypte antique. Leur travail minutieux et leur sens du détail ont grandement contribué à l'immersion du joueur dans notre univers de jeu.

Notre game designer Mana BROWN-VIRTOS a su apporter des idées créatives et assembler les maps de manière ingénieuse. Il a également géré les collisions des différents éléments du jeu sur les cartes, grâce à une classe spécialement rédigée par Noah, qui a grandement simplifié cette tâche complexe. L'assemblage des cartes et la gestion des collisions ont été des éléments clés pour offrir une expérience fluide et agréable pour les joueurs, et c'est grâce à l'expertise de notre équipe que ce défi a pu être relevé.

Au niveau de l'organisation, notre équipe a été confrontée à plusieurs défis. Tout d'abord, le fait que nous avons commencé le projet en mars, juste avant les épreuves de spécialités, a réduit notre temps de travail, car nous avons dû nous concentrer sur nos révisions. En outre, travailler sous la pression de l'examen du baccalauréat aurait été une mauvaise idée. Cependant, malgré ces difficultés, nous avons réussi à rendre le projet à temps grâce aux heures dédiées aux projet dans l'enseignement de la NSI. Nous avons utilisé ce temps pour organiser des réunions et assembler les travaux de chacun. Pour compenser le manque de temps, nous avons travaillé en

dehors de l'établissement, en utilisant un dossier partagé pour faciliter l'échange de fichiers. Nous avons également utilisé une messagerie instantanée pour communiquer efficacement avec tous les membres du groupe.

LES ÉTAPES DU PROJET :

• Présenter les différentes étapes du projet (de l'idée jusqu'à la finalisation du projet)

La première étape était la recherche d'idées pour le projet. Noah avait une approche plus ludique avec l'idée d'un jeu de tir amusant, tandis que Gauthier avait une approche plus éducative en proposant des énigmes liées à la spécialité NSI. Chacun des membres de l'équipe a contribué en ajoutant sa propre touche personnelle et en proposant des idées pour améliorer le jeu. C'est ainsi que nous avons trouvé le thème de l'Égypte antique pour notre jeu.

La deuxième étape a été consacrée à la réalisation du projet. Bien que nous ayons eu des heures en classe pour avancer notre projet, nous avons principalement utilisé ces moments pour discuter et échanger sur le projet. La plupart du travail s'est donc déroulé en dehors de l'école, avec chaque membre de l'équipe exécutant des tâches spécifiques sous la direction de Noah et Gauthier.

Enfin, la dernière étape consistait en l'assemblage final de tous les éléments du projet dans la dernière semaine avant le rendu. Cela impliquait l'assemblage des morceaux de code, des graphismes et des sons, ainsi que la correction des derniers bugs et la rédaction de la documentation. C'est également durant cette période que nous avons tourné notre vidéo et commenté notre code. Nous avons travaillé dur pour mener à bien notre projet et nous sommes fiers du résultat final.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

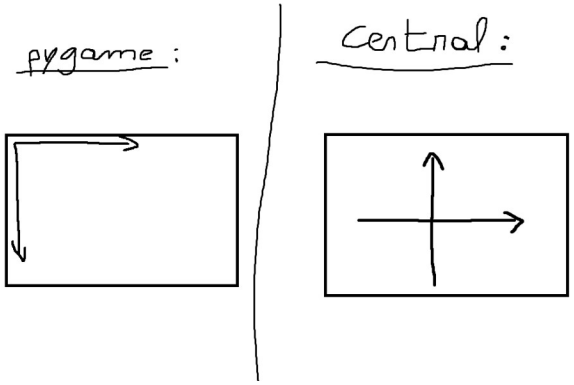
• Avancement du projet (ce qui est terminé, en cours de réalisation, reste à faire)
• Approches mises en œuvre pour vérifier l'absence de bugs et s'assurer de la facilité d'utilisation du projet
• Difficultés rencontrées et solutions apportées

Notre jeu vidéo est actuellement jouable, mais en raison du manque de temps, nous n'avons pas pu intégrer tous les éléments de gameplay que nous avions initialement prévus. Par exemple, nous souhaitions modéliser cinq cartes, mais nous avons finalement pu n'en réaliser que trois. Nous voulions également inclure davantage d'énigmes et varier les ennemis, mais nous avons manquer de temps pour le faire.

Pour garantir la stabilité du projet et réduire le nombre de bugs, nous avons adopté une approche pédagogique en demandant à notre professeur de présenter notre jeu à ses élèves de première, ce qu'il a accepté. Cela nous a permis d'obtenir des retours et des réactions en direct, que vous avez pu voir dans la vidéo (vous pouvez trouver les autorisations de diffusion des élèves de première dans le dossier de doc/interview_autorisations).

Nous avons rencontré de nombreux problèmes, notamment des difficultés liées aux mathématiques. Je vais maintenant vous présenter ces problèmes point par point.

- Repère orthonormé : *La première difficulté que nous avons rencontrée était d'adapter le repère orthonormé de Pygame à celui que nous avons utilisé tout au long de l'année, qui était centré et dont l'axe y était dirigé vers le haut. En effet, dans Pygame, l'origine est placée en haut à gauche de l'écran et l'axe y est dirigé vers le bas. Pour surmonter cette difficulté, nous avons créé une fonction qui convertit les coordonnées du repère centré en coordonnées du repère de Pygame. Cela nous a permis de positionner les éléments de manière plus instinctive. Nous avons également créé une fonction qui fait l'inverse, convertissant les coordonnées du repère de Pygame en coordonnées du repère centré. Cela nous a aidés à déterminer la position de la souris par rapport à notre repère.*



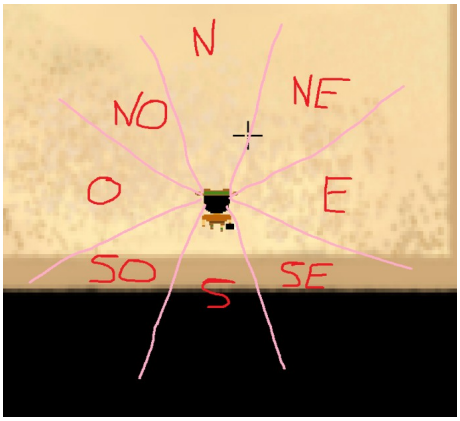

```python
def repaire_orthonorme(x, y, taille_x, taille_y, screen_info = "") :
    """Retourne un tuple qui contient la conversion des valeur d'un repert un en orthonomée centrée à haut à gauche
        repère othonormée centrée -> repère othonormée en haut à gauche
    """
    try  :
        screen_info = pygame.display.Info()

    except pygame.error :
        pass

    return (x + screen_info.current_w / 2 - taille_x / 2, -y + screen_info.current_h / 2 - taille_y / 2)

def repaire_orthonorme_reciproque(x, y, taille_x, taille_y, screen_info = ""):
    """Retourne un tuple qui contient la conversion des valeur d'un repère un en haut à gauche à orthonomée centrée
        repère othonormée en haut à gauche -> repère othonormée centrée
    """
    try  :
        screen_info = pygame.display.Info()

    except pygame.error :
        pass

    return (x - screen_info.current_w / 2 - taille_x / 2, -y + screen_info.current_h / 2 - taille_y / 2)
```

- Vecteurs : *Nous avons également rencontré des difficultés avec les vecteurs. En effet, pour que les balles partent vers le clic de la souris, nous avons dû créer un vecteur directionnel reliant le personnage à la souris. Cela a fonctionné, mais nous avons constaté que plus la souris était éloignée, plus la balle allait vite puisque le vecteur directionnel avait une norme plus grande. Pour résoudre ce problème, nous avons mis en place une fonction qui normalisait les vecteurs en divisant la norme du vecteur directionnel par la distance entre le personnage et la souris. Ainsi, les balles partaient toujours dans la bonne direction et à une vitesse constante.*

```python
def normalize(pos0:Vector2, pos1:Vector2) :
    """Retourne le veteur normalistée avec x et y compris entre -1 et 1"""
    return Vector2(pos1.x - pos0.x, pos1.y - pos0.y) / distance(pos0, pos1)
```

Trigonométrie : *Le mouvement du personnage dans notre jeu n'est pas limité à quatre directions, mais offre huit possibilités en fonction de la position de la souris. Cette fonctionnalité, en apparence anodine, a en réalité nécessité une compréhension approfondie de la trigonométrie. Nous avons effectué des calculs avec des demi-cercles en utilisant la constante pi, avant de les convertir en conditions pour déterminer la position de la souris par rapport au personnage sur huit points cardinaux. Au niveau de la nomenclature, les sprites de notre jeu sont nommés en fonction de leurs positions, leurs directions et leurs numéros de frame. Par exemple, un personnage regardant vers le sud-ouest sur la frame 3 est nommé "persoSO-3".*

```python
def regard_joueur() :
    """Renvoie une string de la direction du joueur en point cardinaux par rapport à un cercle trigonométrique divisé en 8.
    Voir Documentation du jeu."""
    dir_souris = direction_souris()
    x = dir_souris.x
    y = dir_souris.y
    if 0.38 >= x >= -0.38 and y >= 0.92 :
        return "N"
    elif -0.38 >= x >= -0.92 and 0.92 >= y >= 0.38 :
        return "NO"
    elif x <= -0.92 and 0.38 >= y >= -0.38 :
        return "O"
    elif -0.38 >= x >= -0.92 and -0.38 >= y >= -0.92 :
        return "SO"
    elif 0.38 >= x >= -0.38 and -0.92 >= y :
        return "S"
    elif 0.92 >= x >= 0.38 and -0.38 >= y >= -0.92 :
        return "SE"
    elif x >= 0.92 and 0.38 >= y >= -0.38 :
        return "E"
    elif 0.92 >= x >= 0.38 and 0.92 >= y >= 0.38:
        return "NE"
    else :
        return None
```



Collisions : *Pygame dispose d'un système de collision natif, mais nous avons décidé de créer le nôtre pour avoir un contrôle total sur les classes du jeu. Cependant, nous avons rencontré des difficultés pour optimiser cette classe. Nous avons finalement adopté une méthode efficace pour détecter un grand nombre de collisions en parcourant une liste par compréhension et en analysant si le personnage entre en collision avec un objet, en identifiant le côté de contact et en empêchant le joueur d'avancer dans cette direction.*

```
195         if clavier.get(pygame.K_z):  # Le joueur veut aller vers le haut
196             can_bouge = True  # on initialise une variable can_bouge
197             for col in Maps[index_map].liste_collision : # On parcours toutes les collisions de la map
198                 if not Perso.transform.collision.touche(Perso.transform, col).y != 1 :  # Comme le personnage va vers
199                     can_bouge = False  # une des collisions bloque le perso, on l'interdit de bouger.
200             if can_bouge :  # Si il peut bouger alors on le déplace dans la direction souhaité
201                 bouge_tout(Vector2(0, -1))
```

```
83  class Collision :
84      def __init__(self, taille=Vector2(), decalage=Vector2()):
85          self.taille = taille
86          self.decalage = decalage
87
88      def touche(self, objet0:Transform, objet1:Transform) :  # obj 0 étant celui qui appelle la fonction
89          sens_darrivee = Vector2(0, 0)
90          if self.est_dans_zone_x(objet0, objet1) :  # Si l'objet0 est dans la zone x de l'objet1
91              if objet0.position.y > objet1.position.y and self.bord_y(objet1, 1) - 5 <= self.bord_y(objet0, -1) <= self.bord_y(objet1, 1) :  # Si l'objet0 arrive
92                  sens_darrivee.y = -1
93              elif objet0.position.y < objet1.position.y and self.bord_y(objet1, -1) + 5 >= self.bord_y(objet0, 1) >= self.bord_y(objet1, -1) :
94                  sens_darrivee.y = 1
95          if self.est_dans_zone_y(objet0, objet1) :  # Si l'objet0 est dans la zone y de l'objet1
96              if objet0.position.x > objet1.position.x and self.bord_x(objet1, 1) - 5 <= self.bord_x(objet0, -1) <= self.bord_x(objet1, 1) :  # Si l'objet0 arrive
97                  sens_darrivee.x = 1
98              elif objet0.position.x < objet1.position.x and self.bord_x(objet1, -1) + 5 >= self.bord_x(objet0, 1) >= self.bord_x(objet1, -1) :
99                  sens_darrivee.x = -1
100
101         return sens_darrivee  # On retourne le sens par lequel la collision est touché
```

> OUVERTURE :

• Idées d'améliorations (nouvelles fonctionnalités)
• Stratégie de diffusion pour toucher un large public (faites preuve d'originalité !)
• Analyse critique du résultat (si c'était à refaire, que changeriez-vous dans votre organisation, les fonctionnalités du projet et les choix techniques ?)

Comme nous l'avons mentionné précédemment, le manque de temps nous a empêchés de terminer complètement notre jeu. Pour améliorer notre projet, la première chose à faire serait de diversifier le bestiaire des ennemis et des énigmes, car notre jeu actuel peut être complété en une heure, ce qui est assez court pour l'industrie du jeu vidéo. Nous pourrions ainsi ajouter plus de contenu pour prolonger la durée de vie du jeu.

Nous aurions également pu améliorer la présentation du jeu en ajoutant un menu pour donner une meilleure première impression aux joueurs et rendre le démarrage du jeu moins abrupt.

En termes de stratégie de diffusion, nous avons envisagé deux options. Tout d'abord, nous aurions pu créer une présence sur les réseaux sociaux, en particulier sur Instagram, pour montrer en temps réel les progrès de notre projet et attirer l'attention des joueurs potentiels. Ensuite, nous aurions pu publier le jeu sur une plateforme en ligne comme itch.io pour récolter des avis et toucher un public plus large. Cependant, étant donné que nous ne sommes pas dans cette optique pour le moment, ces stratégies ne seront probablement pas mises en œuvre.

Enfin, nous avons appris que la réalisation d'un projet de cette envergure prend du temps. Si nous devions recommencer, nous commencerions bien plus tôt et nous envisagerions d'utiliser un moteur de jeu tel que Unity, qui faciliterait grandement le processus de développement et de production de projets de cette taille.

DOCUMENTATION

• Spécifications fonctionnelles (guide d'utilisation, déroulé des étapes d'exécution, description des fonctionnalités et des paramètres)
• Spécifications techniques (architecture, langages et bibliothèques utilisés, matériel, choix techniques, format de stockage des données, etc)
• Illustrations, captures d'écran, etc

Étapes avant exécution

Avant d'utiliser notre programme, il est important de noter que nous utilisons plusieurs bibliothèques qui ne sont pas natives dans Python, notamment pygame. Afin d'exécuter le programme sans problème, vous devrez installer ces deux bibliothèques sur votre ordinateur. Pour ce faire, ouvrez un terminal Python et tapez les commandes suivantes :

pip install pygame
cd pyracode
python3 Code/main.py

ATTENTION : Certaines bibliotèques tel que copy était intégré à notre version de python, si ce n'est pas le cas vous devriez
taper dans un terminal Python : pip install nom_du_package_manquant

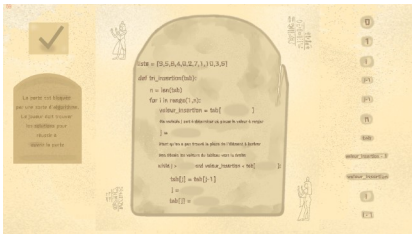
Une fois ces étapes réalisées, amusez-vous !

Comment jouer

Vous avez démarré le jeu, vous devriez tomber sur une magnifique animation dessinée par Léa et doublé par Noah. A la fin de celle ci, le jeu incite le joueur à appuyer sur [Entrée] afin de lancer le jeu.
Notons qu'après ça vous pourrez quitter le jeu en appuyant sur la touche échap.

Ça y est, vous êtes arrivé dans la pyramide ! Faisons un point sur les contrôles du jeu. Les déplacements se font grâce aux touches Z, Q, S et D. La souris vous sera d'une grande utilité dans cette aventure. En effet, en plus de pouvoir viser en la déplaçant et tirer grâce au clique gauche. Elle permet également d'interagir avec certains éléments de la map comme les portes, les socles à énigmes ou encore les clefs.
Vous avez également accès au sac à dos, ouvrable avec M qui permet de refaire les énigmes que vous avez déjà réussies (en cliquant sur le cerveau).
Il vous permet aussi de régler la difficulté des énigmes (en cliquant sur le bras musclé).
Et pour finir vous pourrez quitter le jeu en cliquant sur la croix rouge.

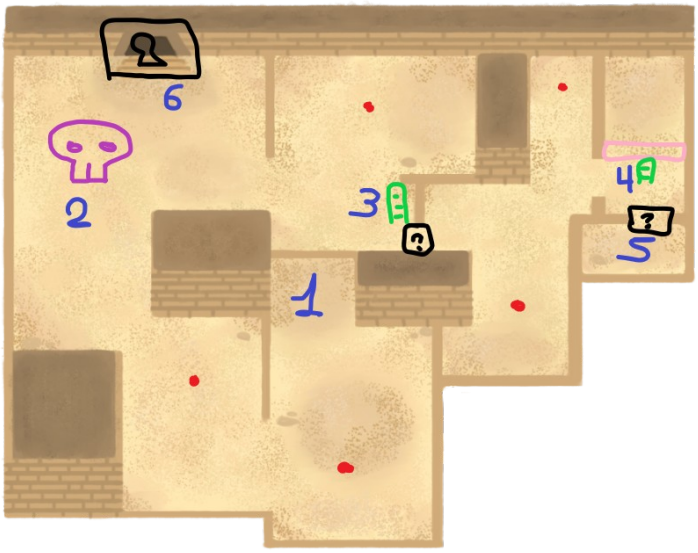

La souris vous permettra également de réaliser les énigmes du jeu comme le code à trou.



Revenons en à la pyramide. Chaque niveau est composé de nombreux éléments de Gameplay. Le but du jeu est de trouver le sarcophage du Pharaon, qui se trouve au dernier étage de la pyramide. Pour cela vous devez trouver la porte qui mène à la salle suivante eouvrable avec une une clé, elle même dissimulé sur des parties de la map accessible en faisant des énigmes.

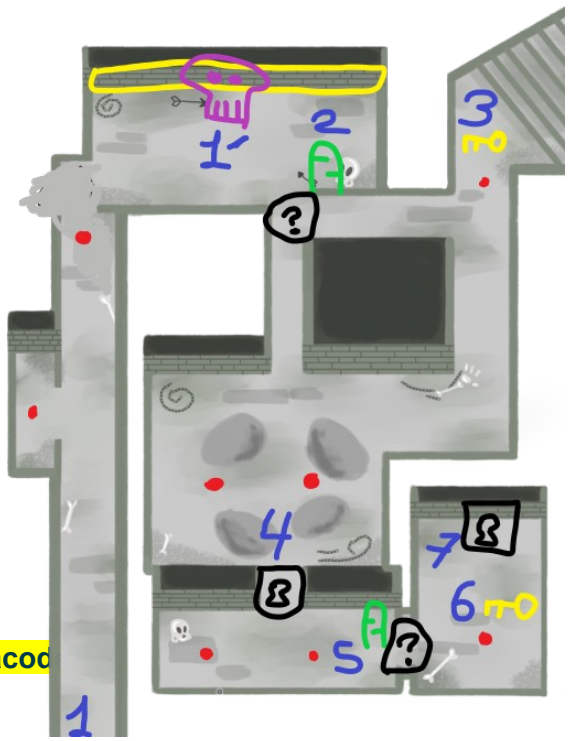
Je vais maintenant vous montrer des schémas avec étape pour vous montrer comment finir le jeu.
Notons qu'il est possible de passer les énigmes si elles sont trop compliqués. Pour cela vous pouvez user du code de triche que nous avons mis en place. Pour l'activer vous rentrez dans une énigme, et vous appuyez sur « R » puis sur « P ». Si vous voulez plus d'information sur les énigmes, allez jeter un œil au dossier explications_énigmes.

Map 1



Voici le déroulé des étapes de la map 1 :
1 : Voici l'endroit où vous apparaissez.
2 : Vous devez éliminer le boss, attention il jette des bombes.
3 : Ici vous devrez réaliser l'énigme afin que la porte s'ouvre. Utilisez le code de triche si elle vous paraît compliqué.
4 : Encore une énigme, réalisez la afin d'ouvrir la porte.
5 : Récupérez la clef.
6 : Ouvrez la porte afin d'accéder à l'étage supérieur.

Map 2



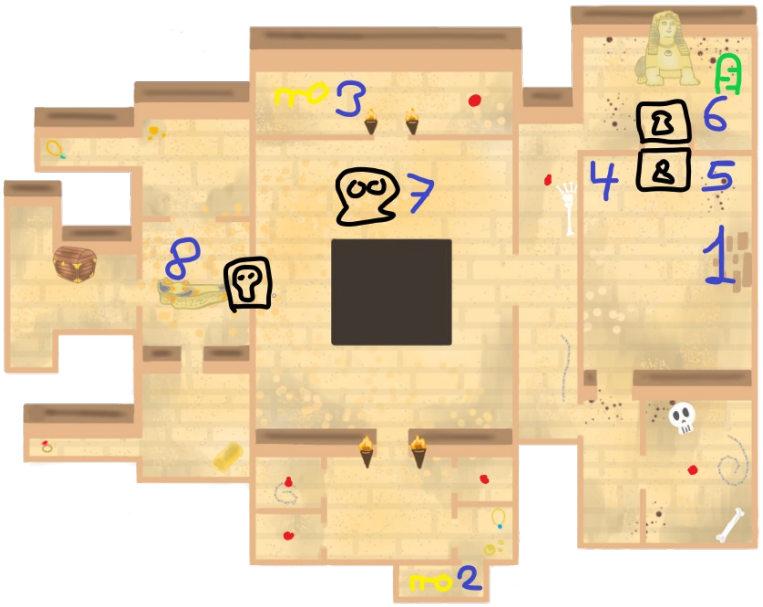
Voici le déroulé des étapes de la map 2 :

1 : Voici l'endroit où vous apparaissez.
1' : Tuez le boss, attention il a un bouclier, il faut donc tirer sur le mur réflectif derrière lui pour faire rebondir les balles et le toucher dans le dos
2 : Résolvez l'énigme pour ouvrir la porte.
3 : Récupérez la clé.
4 : Ouvrez la porte.

Pyracod

5 : Faites l'énigme.
6 : Récupérez la clef.
7 : Ouvrez la porte afin d'arriver à l'étage suivant.

Map 3



Voici le déroulé des étapes de la map 3 :

1 : Voici l'endroit où vous apparaissez.
2 : Prenez la clé du sud.
3 : Prenez la clé du nord.
4 : Ouvrez la première porte.
5 : Ouvrez la deuxième porte.
6 : Faîtes l'énigme du sphinx.
7 : Attention, le grand pharaon veut protéger son tombeau. Il est apparu au milieu de la map. Soyez méfiant car il n'y a pas de stratégie pour le battre. Son point fort est sa cadence de tir extraordinaire. Tuez le, et le moaï gardant l'entrée du tombeau disparaîtra.
8 : Cliquez sur le tombeau, vous avez découvert un secret qui vous vaudra le prix Nobel d'archéologie. Merci d'avoir testé notre projet Pyracode !

Si vous voulez recommencer le jeu, vous avez juste à supprimer le dossier « Sauvegarde » et relancer le jeu.

Spécifications techniques

Dans cette section, nous allons aborder les spécifications techniques de notre projet. Nous avons choisi Python car c'était le seul langage orienté objet enseigné dans le programme d'NSI. Nous avons utilisé le paradigme orienté objet pour l'architecture de notre code, avec un fichier principal "main" qui coordonne l'ensemble des classes codées et réparties dans d'autres fichiers. Nous avons opté pour la bibliothèque 'pygame' pour l'interface graphique car elle permet une création rapide et optimisée. Nous avons également utilisé la bibliothèque 'copy' pour la méthode deepcopy() qui nous a permis d'éviter les problèmes de réplication d'adresses mémoires dans certaines classes.

En plus de ces bibliothèques, nous avons utilisé des bibliothèques natives à Python telles que 'os', qui nous a permis de mettre en place notre système de sauvegarde, et 'math' pour la méthode sqrt() utilisée dans certains calculs de distances.

Pour le matériel, nous avons tous utilisé des ordinateurs, sauf Léa qui a utilisé sa tablette Android et un stylet pour dessiner de manière plus précise.

Concernant les choix de conception, nous avons opté pour un jeu en 2D vu de haut pour éviter de coder la physique avec la gravité. Nous avons fixé la taille de la fenêtre du programme à 1280x720 pixels pour des raisons d'optimisation et de compatibilité avec les écrans actuels. L'adaptation des éléments graphiques à la fenêtre aurait pris trop de temps, nous avons donc opté pour cette résolution fixe.

Nous avons également mis en place un système de sauvegarde efficace en utilisant la méthode native de Python, 'open()', ce qui nous a permis d'écrire et de lire facilement des fichiers texte pour sauvegarder les données du jeu. Cependant, ce système de sauvegarde n'est pas sécurisé, car n'importe qui peut modifier les données puisqu'elles ne sont pas chiffrées ou stockées dans un système de gestion de base de données.

Ce système nous a servis à se souvenir sur quel map était le joueur lorsqu'il a quitté le jeu, ou encore qu'elles énigmes il a réalisé pour qu'il puisse les refaire à partir de sons sac à dos.

Nous vous remercions d'avoir lu cette documentation. Cordialement, l'équipe de Pyracode.